Householder transformation

(Algorithm)

The most frequently applied algorithm for QR decomposition uses the Householder transformation $u = Hv$, where the Householder matrix H is a symmetric and orthogonal matrix of the form:

$$H = I - 2xx^T$$

with the identity matrix I and any normalized vector x with $||x||_2^2 = x^T x = 1$.

Householder transformations zero the $m-1$ elements of a column vector v below the first element:

$$\begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{bmatrix} \rightarrow \begin{bmatrix} c \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \text{with } c = \pm ||v||_2 = \pm \left(\sum_{i=1}^{m} v_i^2 \right)^{1/2}$$

One can verify that

$$x = f \begin{bmatrix} v_1 - c \\ v_2 \\ \vdots \\ v_m \end{bmatrix} \quad \text{with } f = \frac{1}{\sqrt{2c(c - v_1)}}$$

fulfils $x^T x = 1$ and that with $H = I - xx^T$ one obtains the vector $\begin{bmatrix} c & 0 & \cdots & 0 \end{bmatrix}^T$.

To perform the decomposition of the $m \times n$ matrix $A = QR$ (with $m \geq n$)

Classification
News
Legalese
History
ChangeLog
TODO List

we construct in this way an $m \times m$ matrix $H^{(1)}$ to zero the $m-1$ elements of the first column. An $m-1 \times m-1$ matrix $G^{(2)}$ will zero the $m-2$ elements of the second column. With $G^{(2)}$ we produce the $m \times m$ matrix

$$H^{(2)} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & & & \\ \vdots & & G^{(2)} & \\ 0 & & & \end{bmatrix}, \quad \text{etc}$$

After n ($n-1$ for $m=n$) such orthogonal transforms $H^{(i)}$ we obtain:

$$R = H^{(n)} \cdots H^{(2)} H^{(1)} A$$

R is upper triangular and the orthogonal matrix Q becomes:

$$Q = H^{(1)} H^{(2)} \cdots H^{(n)}$$

In practice the $H^{(i)}$ are never explicitely computed.

References

- Originally from The Data Analysis Briefbook (http://rkb.home.cern.ch/rkb/titleA.html)

"Householder transformation" is owned by akrowne.

(view preamble)

View style: HTML with images reload

See Also: Gram-Schmidt orthogonalization

Other names: Householder reflection, Householder matrix

Keywords: matrix orthogonalization

Cross-references: upper triangular, orthogonal, column, matrix, decomposition, column vector, vector, identity matrix, orthogonal matrix, symmetric, QR decomposition
There are 5 references to this object.

This is version 3 of Householder transformation, born on 2002-01-04, modified 2002-03-08.
Object id is 1210, canonical name is HouseholderTransformation.
Accessed 4858 times total.

Classification:

AMS MSC: 65F25 (Numerical analysis :: Numerical linear algebra :: Orthogonalization)
15A57 (Linear and multilinear algebra; matrix theory :: Other types of matrices)

Pending Errata and Addenda

None.

Discussion

Style: Threaded Expand: 1 Order: Newest first reload

No messages.

Interact

rate | post | correct | update request | add example | add (any)

Next Up Previous

Next: Performance Results **Up:** Factorization Routines **Previous:** QR Factorization

Cholesky Factorization

Cholesky factorization factors an $N \times N$, symmetric, positive-definite matrix A into the product of a lower triangular matrix L and its transpose, i.e., $A = LL^T$ (or $A = U^T U$, where U is upper triangular). It is assumed that the lower triangular portion of A is stored in the lower triangle of a two-dimensional array and that the computed elements of L overwrite the given elements of A. At the k-th step, we partition the $n \times n$ matrices $A^{(k)}$, $L^{(k)}$, and $L^{(k)^T}$, and write the system as

$$\begin{pmatrix} A_{11} & A_{21}^T \\ A_{21} & A_{22} \end{pmatrix} = \begin{pmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{pmatrix} \begin{pmatrix} L_{11}^T & L_{21}^T \\ 0 & L_{22}^T \end{pmatrix}$$
$$= \begin{pmatrix} L_{11} L_{11}^T & L_{11} L_{21}^T \\ L_{21} L_{11}^T & L_{21} L_{21}^T + L_{22} L_{22}^T \end{pmatrix}$$

where the block A_{11} is $n_b \times n_b$, A_{21} is $(n - n_b) \times n_b$, and A_{22} is $(n - n_b) \times (n - n_b)$. L_{11} and L_{22} are lower triangular.

The block-partitioned form of Cholesky factorization may be inferred inductively as follows. If we assume that L_{11}, the lower triangular Cholesky factor of A_{11}, is known, we can rearrange the block equations,

$$L_{21} \leftarrow A_{21}(L_{11}^T)^{-1},$$
$$\tilde{A}_{22} \leftarrow A_{22} - L_{21} L_{21}^T = L_{22} L_{22}^T.$$

A snapshot of the block Cholesky factorization algorithm in Figure 5 shows how the column panel $L^{(k)}$ (L_{11} and L_{21}) is computed and how the trailing submatrix A_{22} is updated. The factorization can be done by recursively applying the steps outlined above to the $(n - n_b) \times (n - n_b)$ matrix $\tilde{A}_{22}$.

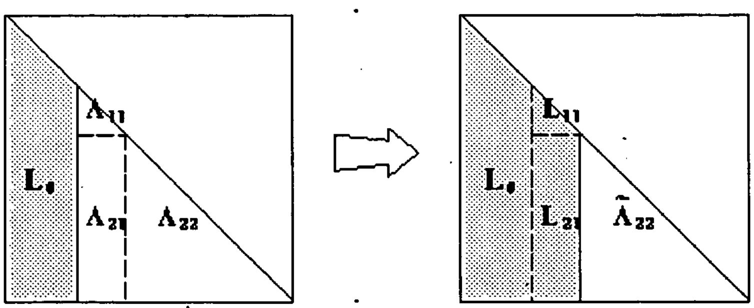

Figure 5: A snapshot of block Cholesky factorization.

In the right-looking version of the LAPACK routine, the computation of the above steps involves the following operations:

1. DPOTF2: Compute the Cholesky factorization of the diagonal block A_{11}.

   $$A_{11} \rightarrow L_{11} L_{11}^T$$

2. DTRSM: Compute the column panel L_{21},

   $$L_{21} \leftarrow A_{21}(L_{11}^T)^{-1}$$

3. DSYRK: Update the rest of the matrix,

   $$\tilde{A}_{22} \leftarrow A_{22} - L_{21} L_{21}^T = L_{22} L_{22}^T$$

The parallel implementation of the corresponding ScaLAPACK routine, PDPOTRF, proceeds as follows:

1. PDPOTF2: The process P_i, which has the $n_b \times n_b$ diagonal block A_{11}, performs the Cholesky factorization of A_{11}.
   - P_i performs $A_{11} \rightarrow L_{11} L_{11}^T$, and sets a flag if A_{11} is not positive definite.
   - P_i broadcasts the flag to all other processes so that the computation can be stopped if A_{11} is not positive definite.
2. PDTRSM: L_{11} is broadcast columnwise by P_i down all rows in the current column of processes, which computes the column of blocks of L_{21}.
3. PDSYRK: the column of blocks L_{21} is broadcast rowwise across all columns of processes and then transposed. Now, processes have their own portions of L_{21} and L_{21}^T. They update their local portions of the matrix A_{22}.

Next Up Previous

Next: Performance Results **Up:** Factorization Routines **Previous:** QR Factorization

---

Susan Ostrouchov
Fri Apr 28 09:37:26 EDT 1995